



UNIVERSIDAD DEL VALLE  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN  
DESARROLLO DE SOFTWARE II  
Febrero - Junio de 2015

## Taller No. 5

# JUnit y Selenium para Pruebas Unitarias y Funcionales

Preparado por:

Mauricio Castillo Mina 1226715

Walter Yangana Medina 1225941

Revisado Docente:

Beatriz Eugenia Florián Gaviria

Monitora:

Lisa Fernanda Betancourt Escobar

## Instrucciones de Entrega del Taller:

- El taller se podrá realizar en grupos de mínimo 2 estudiantes y máximo 4
- La fecha límite de entrega es el día 23/05/2015 a las 23:55
- Se deben entregar el documento que soporta el desarrollo del taller con pantallazos paso a paso de lo realizado para ambos ejercicios. Además, el proyecto de NetBeans que corresponde al desarrollo del primer ejercicio y por otra parte, el script de los casos de prueba del segundo ejercicio.
- La entrega se realizará únicamente mediante la asignación creada en el campus virtual.

## Contenido

Instrucciones de Entrega del Taller: .....	1
INTRODUCCIÓN .....	2
¿Qué es JUnit?: .....	2
¿Qué es Selenium?:.....	2
PARTE 1: UTILIZANDO JUNIT (3.0 PUNTOS).....	3
Instalación del Entorno de Pruebas .....	3
Ejercicio Guiado (1.0 Puntos).....	5
Exploración del Entorno de Junit en NetBeans .....	5
Aplicando la Técnica de Cobertura de Sentencias:.....	8
Algunos Métodos que usa JUnit para las pruebas: .....	10
Casos de prueba implementados: .....	12
Ejercicio no guiado (2.0 Puntos) .....	12
PARTE 2: SELENIUM IDE (2.0 PUNTOS) .....	14
Instalar Selenium IDE.....	14

Exploración del Entorno de Pruebas.....	17
Utilizando la Técnica de Partición de Equivalencia.....	20
Caso práctico .....	20
Clases de Equivalencia: .....	21
Resumen de caso de prueba: Datos de Entrada y Valor Esperado:.....	22
Demostración de scripts de casos de prueba.....	22
Ejercicio no guiado .....	22
Referencias.....	24

## INTRODUCCIÓN

### **¿Qué es JUnit?:**

JUnit es un conjunto de bibliotecas utilizadas para hacer pruebas unitarias para Java. Fue desarrollada por Eric Gamma y Kent Beck.

JUnit permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera.

Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente.

JUnit es también un medio de controlar las pruebas de retroceso, necesarias cuando una parte del código ha sido modificado y se desea ver que el nuevo código cumple con los requerimientos anteriores y que no se ha alterado su funcionalidad después de la nueva modificación.



Página Web: <http://junit.org/>

### **¿Qué es Selenium?:**

Selenium es un conjunto de utilidades que facilita la labor de obtener juegos de pruebas para aplicaciones web. Para ello nos permite grabar, editar y depurar casos de prueba, que podrán ser ejecutados de forma automática e iterativa posteriormente.

Además de ser una herramienta para registrar acciones, permite editarlas manualmente o crearlas desde cero. Las acciones se basan en el uso de diferentes API's en diferentes lenguajes (PHP, Ruby, JAVA, Javascript, etc).



Página Web: <http://www.seleniumhq.org/>

## PARTE 1: UTILIZANDO JUNIT (3.0 PUNTOS)

### Instalación del Entorno de Pruebas

NetBeans trae por defecto instalado el plug-in para JUnit. En caso de no tenerlo procedemos a descargarlo e instalarlo desde la página oficial de JUnit:

<https://github.com/junit-team/junit/wiki/Download-and-Install> (Ver **¡Error! No se encuentra el origen de la referencia.**).

A screenshot of a web browser displaying the "Download and Install" page for the JUnit repository on GitHub. The URL in the address bar is https://github.com/junit-team/junit/wiki/Download-and-Install. The page title is "GitHub - junit-team / junit". Below the title, there is a section titled "Download and Install" with a note that Anders Pitman edited the page on 24 Jan · 18 revisions. A sub-section titled "Plain-old JAR" provides instructions to download the JAR files and put them on the test classpath. It lists two items: "junit.jar" and "hamcrest-core.jar", with "junit.jar" circled in red. Another sub-section titled "Maven" is also visible on the page.

Figura 1 Página de descarga de la Librería JUnit.

Una vez descargado el plug-in junit.jar (Para nuestro caso instalaremos Junit-12.4.jar) se da clic en el menú de Herramientas/Tools y se elige la opción Libraries, nos saldrá un cuadro de dialogo (Ver Figura 2)

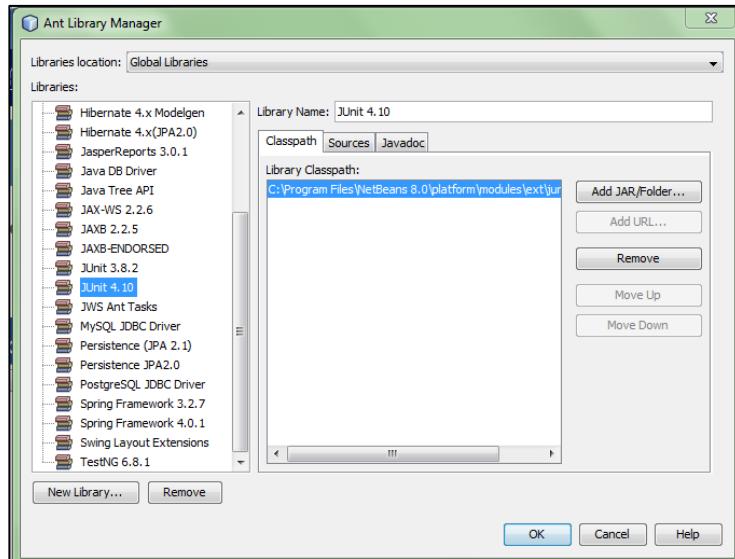


Figura 2 Cuadro de Dialogo Librerías

A continuación damos clic en el botón **New Library**, asignamos un nombre a la librería, para nuestro ejemplo la llamaremos JUnit-12.4.jar, clic en OK. Luego vamos al botón Add JAR/Folder y buscamos el plug-in que descargamos, por último clic en OK. Si las instrucciones se realizaron correctamente en la parte izquierda de la ventana de NetBeans en la carpeta **Test Libraries** se visualizara la librería creada (Ver Figura 3).

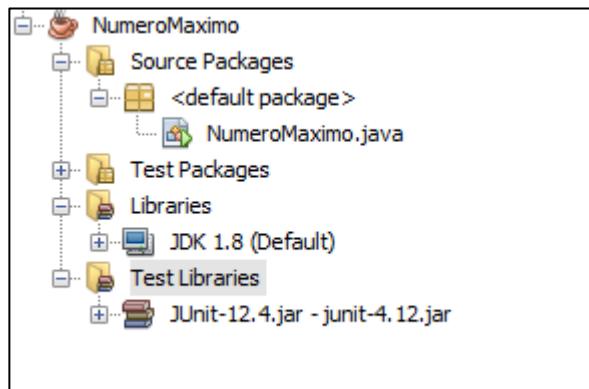


Figura 3 Librería JUnit-12.4.jar adicionada al proyecto de NetBeans.

## Ejercicio Guiado (1.0 Puntos)

### Exploración del Entorno de Junit en NetBeans

Para realizar un primer ejercicio de pruebas unitarias con JUnit crearemos un proyecto en NetBeans y realizaremos una clase llamada **MaximoNumero.java** que contiene un sencillo código como se muestra en la figura 4(El código fuente se encuentra en la carpeta adjunta al documento del taller en el campus).

```
public class NumeroMaximo
{
    public static int calcularMaximo(int x, int y, int z)
    {
        int max=0;

        if (x>y && x>z)
        {
            max = x;
        }

        else
        {
            if (z>y)
            {
                max = z;
            }

            else
            {
                max = y;
            }
        }
        return max;
    }
}
```

Figura 4 Código de la clase **NumeroMaximo.java**

Se selecciona la clase a la que se le quieren hacer pruebas, para el ejemplo tenemos la clase llamada **NumeroMaximo.java**, se da clic en el menú de Herramientas/Tools en NetBeans y se elige la opción Create /Update Tests (Ver Figura 5).

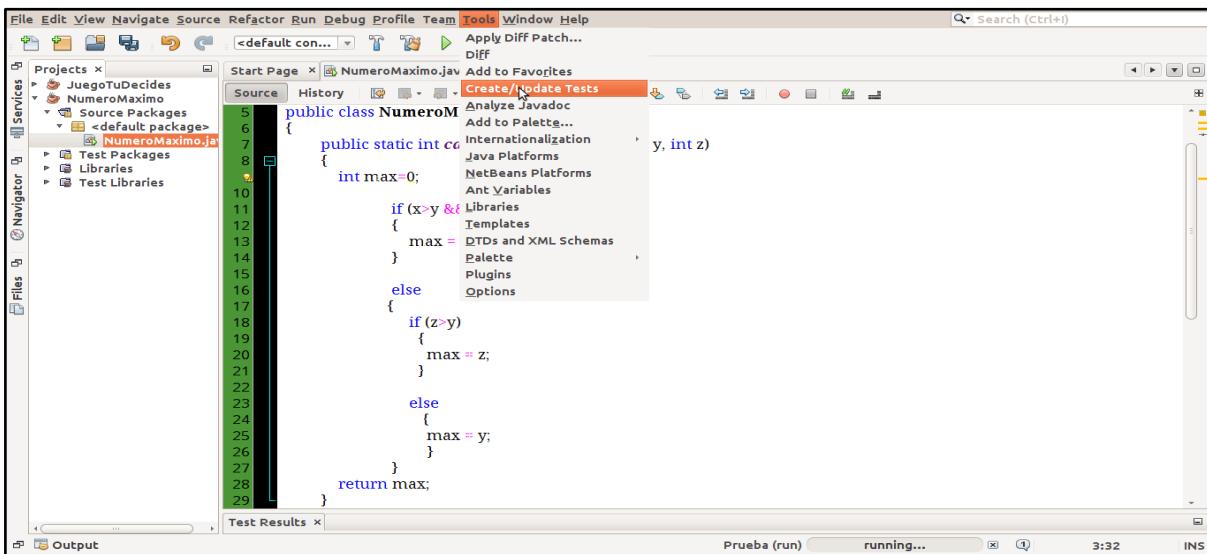


Figura 5 Crear Pruebas

Se elige a JUnit como framework para hacer las pruebas y se deja la configuración por defecto como se muestra en la [¡Error! No se encuentra el origen de la referencia..](#)

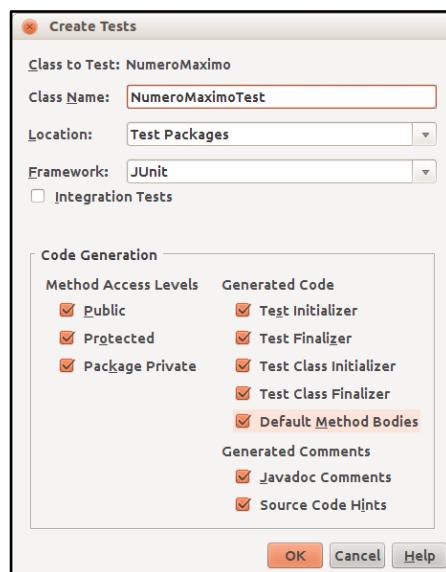


Figura 6 Configuración de las Pruebas

Ahora en la carpeta **Test Package** se debe haber creado la clase **NumeroMaximoTest.java**, en la cual se ha implementado los métodos `testCalcularMaximo()`, `testMain()` (Ver Figura 7).

```

43 public void testCalcularMaximo() {
44     System.out.println("calcularMaximo");
45     int x = 0;
46     int y = 0;
47     int z = 0;
48     int expResult = 0;
49     int result = NumeroMaximo.calcularMaximo(x, y, z);
50     assertEquals(expResult, result);
51     // TODO review the generated test code and remove the default call to fail.
52     fail("The test case is a prototype.");
53 }
54 /**
55 * Test of main method, of class NumeroMaximo.
56 */
57 Test
58 public void testMain() {
59     System.out.println("main");
60     String[] args = null;
61     NumeroMaximo.main(args);
62     // TODO review the generated test code and remove the default call to fail.
63     fail("The test case is a prototype.");
64 }
65 }
66 }
67 }

```

**Figura 7 Implementación de la clase NumeroMaximoTest.java**

Ahora ejecutamos la clase **NumeroMaximoTest.java** y nos mostrará que ninguno de los dos métodos ha pasado la prueba (ver Figura 8), esto se debe a que el método **fail** impide que determinado método supere el test, si comentamos esta línea en los dos métodos, la prueba pasara al 100%, (ver Figura 9).

Test	Status	Message
numeromaximo.NumeroMaximoTest	0,00 %	No test passed, 2 tests failed.(7,154 s)
testCalcularMaximo	Failed	Failed: The test case is a prototype.
testMain	Failed	Failed: The test case is a prototype.

**Figura 8 Falla de los test de prueba**

The screenshot shows an IDE interface with two main parts. The top part is a code editor with the following Java test code:

```

44     Test
45     public void testCalcularMaximo() {
46         System.out.println("calcularMaximo");
47         int x = 0;
48         int y = 0;
49         int z = 0;
50         int expResult = 0;
51         int result = NumeroMaximo.calcularMaximo(x, y, z);
52         assertEquals(expResult, result);
53         // TODO review the generated test code and remove the default call to fail.
54         //fail("The test case is a prototype.");
55     }

```

The bottom part is a "Test Results" window titled "numeromaximo.NumeroMaximoTest". It displays the following information:

- A green progress bar at the top indicating "100,00 %".
- The message "Both tests passed.(5,185 s)".
- A column on the right labeled "main" which contains the word "calcularMaximo".
- Icons for refresh, run, and stop.

Figura 9 Pruebas exitosas

### Aplicando la Técnica de Cobertura de Sentencias:

1. Seleccionamos el método al cual se le hará la prueba en nuestro caso el método calcularMaximo(int x, int y, int z):

```

public static int calcularMaximo(int x, int y, int z){
    int max=0;
        if (x>y && x>z) {
            max = x;
        }
        else {
            if (z>y) {
                max = z;
            } else {
                max = y;
            }
        }
    return max;
}

```

2. Dibujamos el Diagrama de Flujo (Ver figura 10):

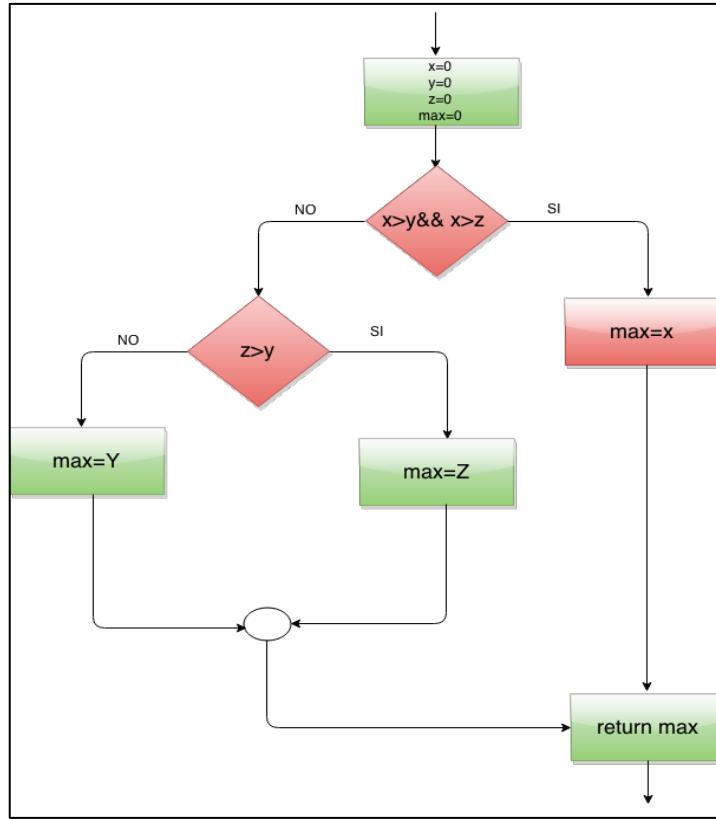


Figura 10 Diagrama de Flujos

3.Identificamos los Caminos (Ver Figura 11):

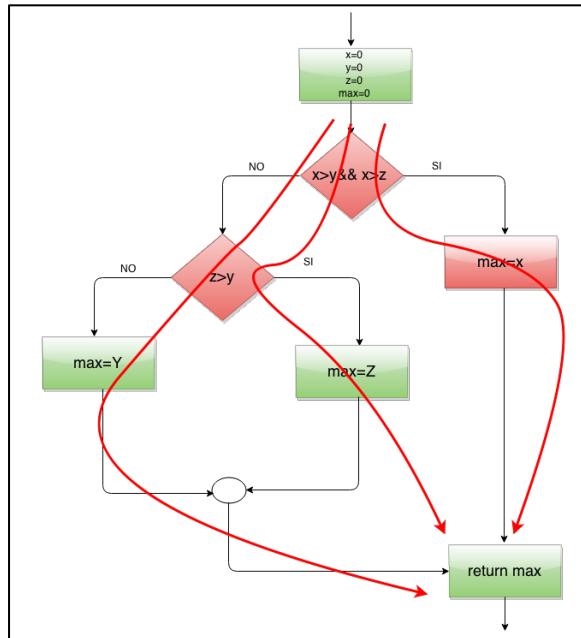


Figura 11 Caminos Identificados.

Hay 3 caminos.

4. Casos de Prueba asociados a los caminos encontrados:

Camino	Caso de prueba	resultado esperado
1	X=6 Y=3 Z=2	AccertEquals max=6
2	X=2 Y=4 Z=10	AccertEquals max=10
3	X=1 Y=5 Z=4?	AccertEquals max=5

Algunos Métodos que usa JUnit para las pruebas:

Método setUp () :

Podremos encontrar casos en los que necesitemos ejecutar algún tipo de tarea específica, antes de lanzar la prueba. Por ejemplo vendría muy bien, para abrir una base de datos antes de comenzar las pruebas. Pasaremos a hacer un ejemplo básico viendo el código, para una mejor comprensión (Ver Figura 12):

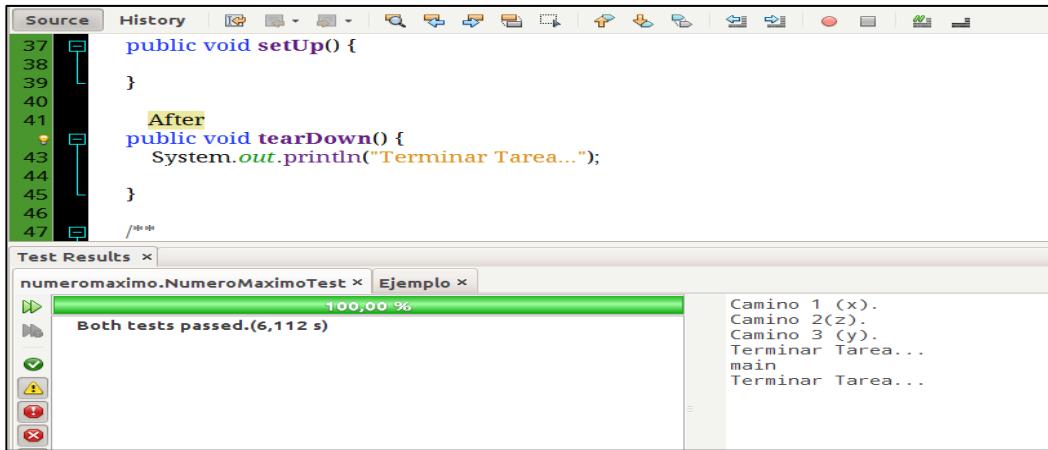
The screenshot shows an IDE interface with two main parts. On the left, there is a code editor displaying Java code. Lines 35 to 45 are visible, showing a 'Before' block with a `setUp()` method that prints "Ejecutar Tarea..." and an 'After' block with a `tearDown()` method. On the right, there is a "Test Results" window titled "numeromaximo.NumeroMaximoTest x Ejemplo x". It shows a green progress bar at 100.00% completion with the message "Both tests passed.(5,222 s)". Below the progress bar, there is a log output: "Ejecutar Tarea... Camino 1 (x). Camino 2(z). Camino 3 (y). Ejecutar Tarea... main". The bottom of the window has standard JUnit icons for run, stop, and refresh.

Figura 12 Método `setUp ()`.

Como se ve en la imagen el método `setUp` comienza ejecutando la tarea contenida en el método `testCalcularMaximo`, para nuestro caso los caminos 1,2 y 3. Seguidamente se termina la ejecución y vuelve a ejecutarse la tarea contenida en el método `Main`.

Método tearDown () :

Este método tiene una función similar a `setUp ()`, con la diferencia de que éste se ejecutará al final del código, para finalizar el tipo de tarea especificada (Ver Figura 13).



**Figura 13 Método tearDown().**

La imagen muestra cómo se ejecuta la instrucciones que están en el método `testCalcularMaximo`, por medio del método `tearDown` se termina esa tarea y entra al método `Main`, ejecuta las respectivas instrucciones del `Main` para luego terminar la tarea.

#### Método assertEquals():

Es un método que compara un resultado esperado, con el resultado calculado por el programa (Ver Figura 14).

```

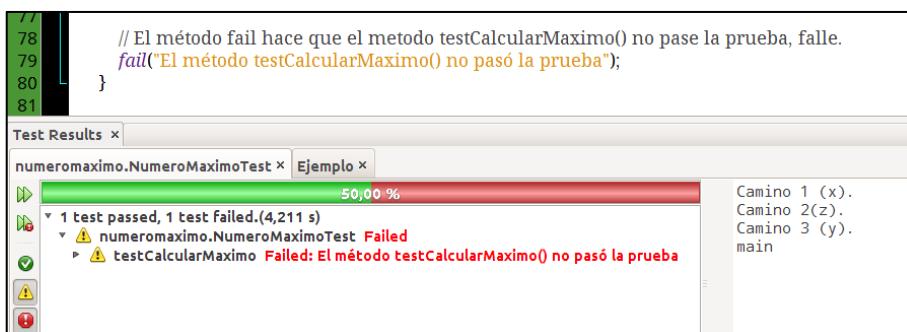
public void testCalcularMaximo() {
    System.out.println("Camino 1 (x).");
    int x = 6;
    int y = 3;
    int z = 2;
    int expResult = x;
    int result = NumeroMaximo.calcularMaximo(x, y, z);
    assertEquals("Resultado Esperado 6",expResult, result);
}

```

**Figura 14 Método assertEqual ().**

#### Método fail():

Este hace que determinada instrucción falle a propósito (Ver Figura 15).



**Figura 15 Método fail () .**

## Casos de prueba implementados:

Caso de Prueba para el camino 1 (x=6, y=3, z=2) resultado 6.

Caso de Prueba para el camino 2 (x=1, y=5, z=4) resultado 5

Caso de Prueba para el camino 3 (x=2, y=4, z=10) resultado 10

(Ver Figura 16).

The screenshot shows an IDE interface with two main panes. The left pane displays a Java test class named `Test` with a single method `testCalcularMaximo()`. The code sets up three variables `x`, `y`, and `z` to values 6, 3, and 2 respectively, calculates their maximum using a `calcularMaximo` method from a `NumeroMaximo` class, and asserts that the result is 6. The right pane shows the `Test Results` window with a green bar indicating 100.00% completion and the message "Both tests passed.(9,263 s)". Below the bar, the terminal output shows the execution of the test and the results of the three paths: Camino 1 (x=6, y=3, z=2) resulting in 6, Camino 2 (x=1, y=5, z=4) resulting in 5, and Camino 3 (x=2, y=4, z=10) resulting in 10.

```
50
51     Test
52     public void testCalcularMaximo() {
53         System.out.println("Camino 1 (x):");
54         int x = 6;
55         int y = 3;
56         int z = 2;
57         int expResult = x;
58         int result = NumeroMaximo.calcularMaximo(x, y, z);
59         assertEquals("Resultado Esperado 6",expResult, result);
60         System.out.println("El número mayor es: "+x);
```

Test Results x

numeromaximo.NumeroMaximoTest

100,00 %

Both tests passed.(9,263 s)

Ejecutar Tarea...  
Camino 1 (x):  
El número mayor es: 6  
Camino 2 (y):  
El número mayor es: 5  
Camino 3 (z):  
El número mayor es: 10  
Terminar Tarea...  
Ejecutar Tarea...  
main  
Terminar Tarea...

Figura 16 Caso se prueba implementado.

Al ejecutar el test miramos que los resultados para cada camino son los esperados.

**Observación:** Junto a este documento se anexa código fuente del Caso de Prueba Implementado.

## Ejercicio no guiado (2.0 Puntos)

Una empresa colombiana desea automatizar el pago de nómina para sus **empleados**, usted debe crear una aplicación con su respectiva interface donde se implemente y pruebe los siguientes métodos de una clase llamada **nómina**.

**Para determinar los casos de pruebas usando la técnica de cobertura de sentencias para al menos 5 métodos de los siguientes.** (ver diapositivas del docente sobre este tema).

### Métodos a implementar:

- **Calcular Sueldo Bruto:** Recibe un empleado y computa su sueldo bruto,(salario sin ningún tipo de descuento).

- **CalcularSueldoReal:** Recibe un empleado y computa su sueldo una vez se le ha descontado el 8% del mismo que es aportado supuestamente a su salud y pensión.
- **CalcularSueldosEmpleados:** Recibe un arreglo de empleados y produce un arreglo con los salarios reales de cada empleado.
- **CalcularSumaSueldos:** Recibe una lista de empleados y computa la suma de todos los salarios reales de los empleados.
- **CalcularPromedioSueldos:** Recibe un arreglo de empleados y computa el promedio de los salarios reales de estos.
- **ObtenerIdEmpleados:** Recibe un arreglo de empleados y retorna un ArrayList con las cédulas de todos los empleados.
- **CalcularTotalAPagar:** Recibe una lista de empleados y computa la suma de todos los salarios en bruto de los empleados.
- **obtenerEmpleadoGanaMenos:** Recibe una lista de empleados y devuelve el que tenga el menor sueldo.

## PARTE 2: SELENIUM IDE (2.0 PUNTOS)



Sitio Web oficial: <http://www.seleniumhq.org/>

### Selenium WebDriver



- crear, suites de automatización de regresión robustas basadas en navegación y pruebas.
- escalar y distribuir secuencias de comandos a través de muchos entornos.

### Selenium IDE



- crear secuencias de comandos para la detección de bug rápidos
- crear scripts para ayudar en la automatización de pruebas exploración.

Selenium IDE es un plugin de Firefox que pertenece al juego de herramientas SeleniumHQ, permite realizar juegos de pruebas sobre aplicaciones web. Para ello realiza la grabación de la acción seleccionada (navegación por una página) en un "script", el cual se puede editar y parametrizar para adaptarse a los diferentes casos, y lo que es más importante su ejecución se puede repetir múltiples veces.

### Instalar Selenium IDE

Para la instalación de Selenium IDE se requiere tener el navegador Firefox instalado.

#### Paso 1:

Ir a esta URL <http://www.seleniumhq.org/download/> donde entrara el instalador de Selenium IDE para Firefox.

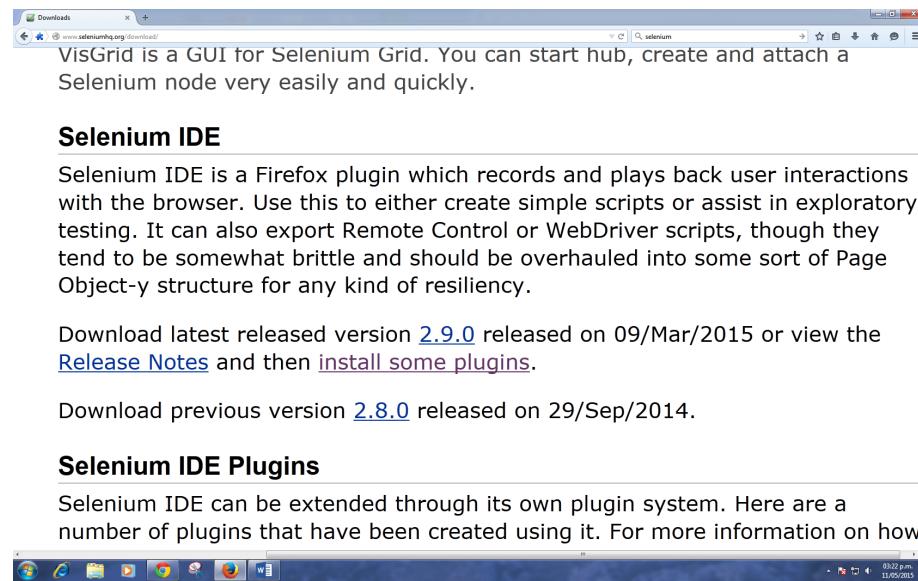


Figura 17

O directamente pulsar el enlace del plugin

<http://release.seleniumhq.org/selenium-ide/2.9.0/selenium-ide-2.9.0.xpi>

**Paso 2:** Instalar el plugin presionándole el botón permitir. Que aparecerá en la parte superior del navegador.

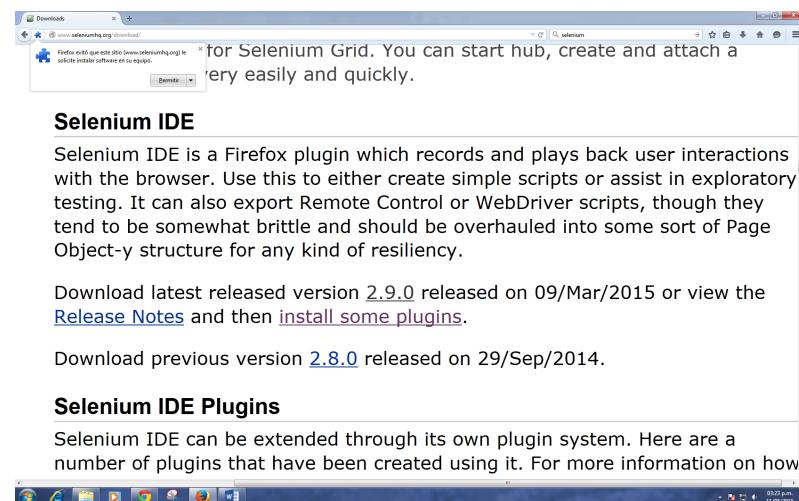


Figura 18

**Paso 3:** Instalamos los complementos y reiniciamos el navegador.

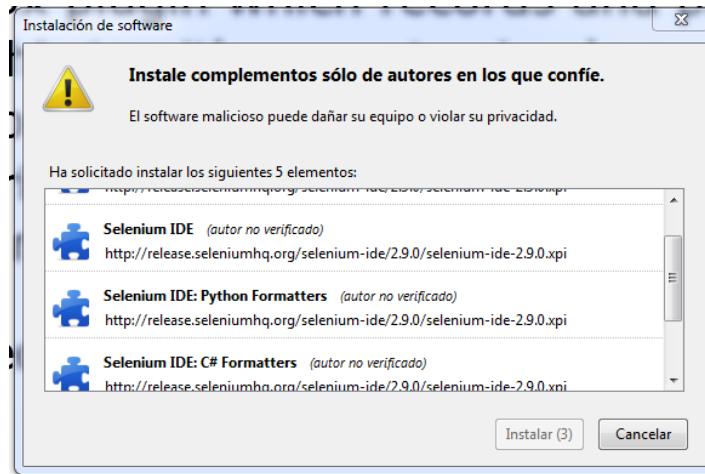


Figura 19

**Paso 4:** abrimos el IDE de Selenium presionando la tecla Alt en el menú herramientas Y Selenium IDE

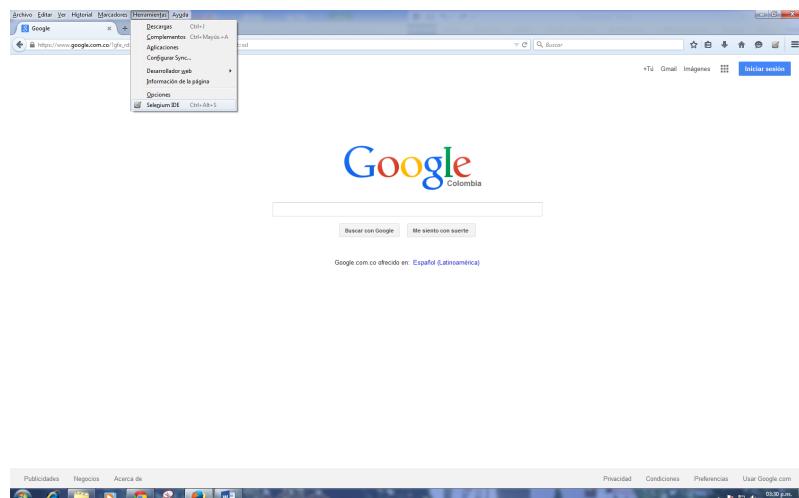


Figura 20

**Paso 5:** Inserción de comandos de pruebas de pruebas.

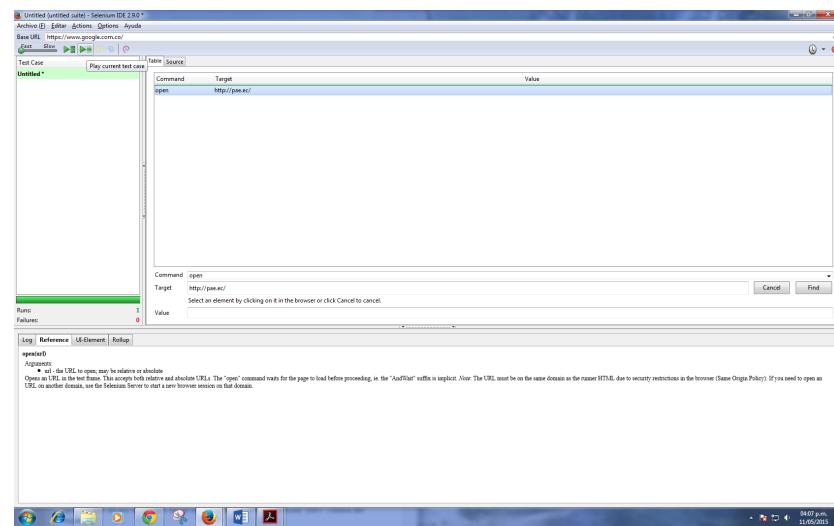


Figura 21

## Exploración del Entorno de Pruebas Instrucciones de Selenium IDE

Command

Target

open

**http://pae.ec/ → Prueba de apertura del sitio web.**

verifyTextPresent

**Protección Animal → Prueba de existencia de texto.**

verifyTextPresent

**Copyright 2011 → Prueba de existencia de texto.**

verifyXPathCount

**//table//tr → Se le anexa un valor para determinar el número de registros mostrados.**

## Paso 6: Creación de Scripts de navegación.

- Activaremos el plugin Selenium

IDE Alt → Herramientas -> Selenium IDE)

En Base URL escribimos <https://www.google.com.co/>



Figura 22

Y presionamos el botón de grabar

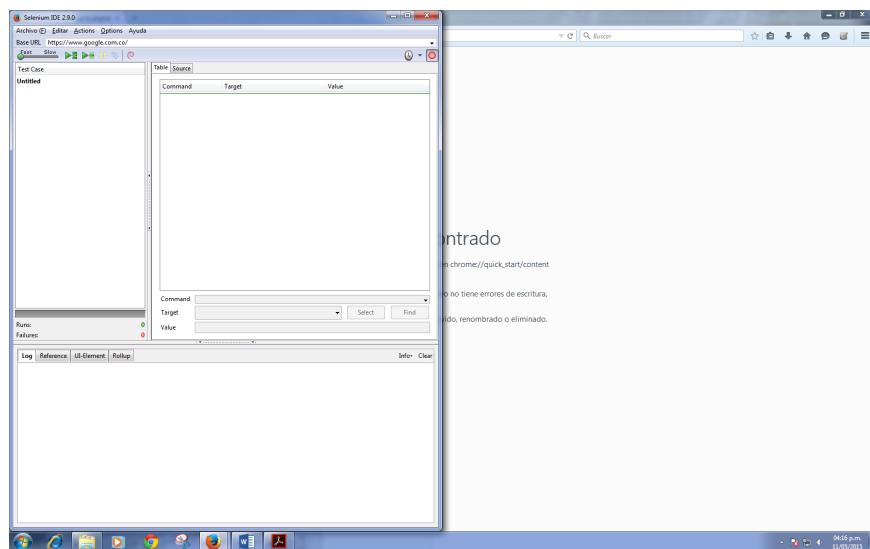


Figura 23

Realizamos la búsqueda en google

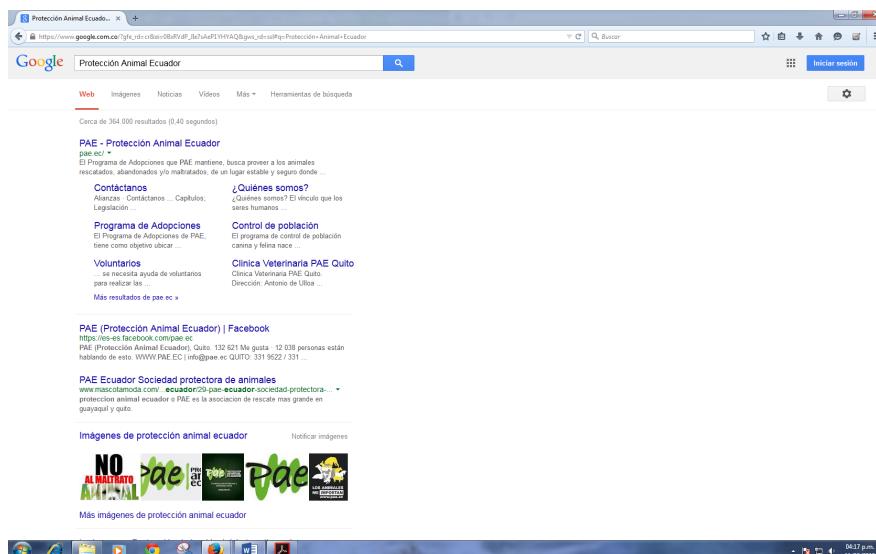


Figura 24

Ingresamos a la pagina.

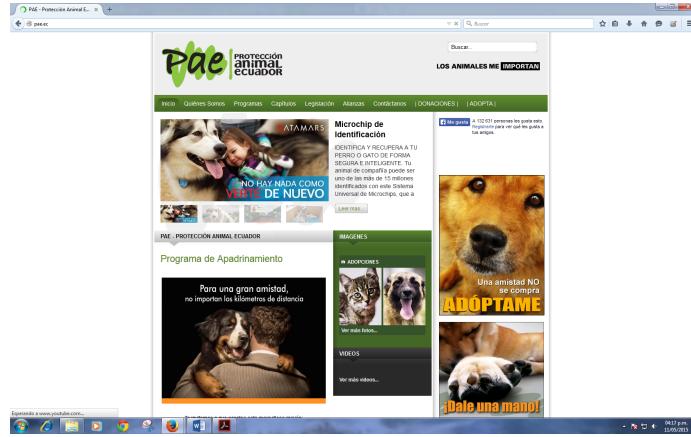


Figura 25

Nos dirigimos a la sección de quien somos

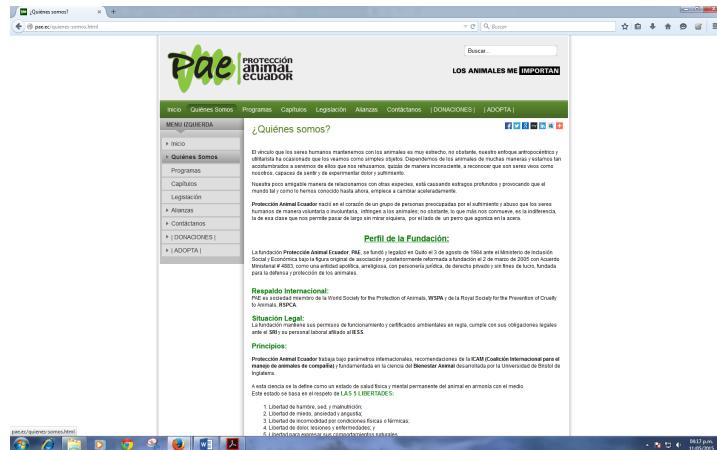


Figura 26

Buscamos en google “Protección Animal Ecuador” le damos al link de que hace referencias al <http://pae.ec/> y nos vamos a la sección de quienes somos.

Recorrido grabado de las pruebas realizadas.

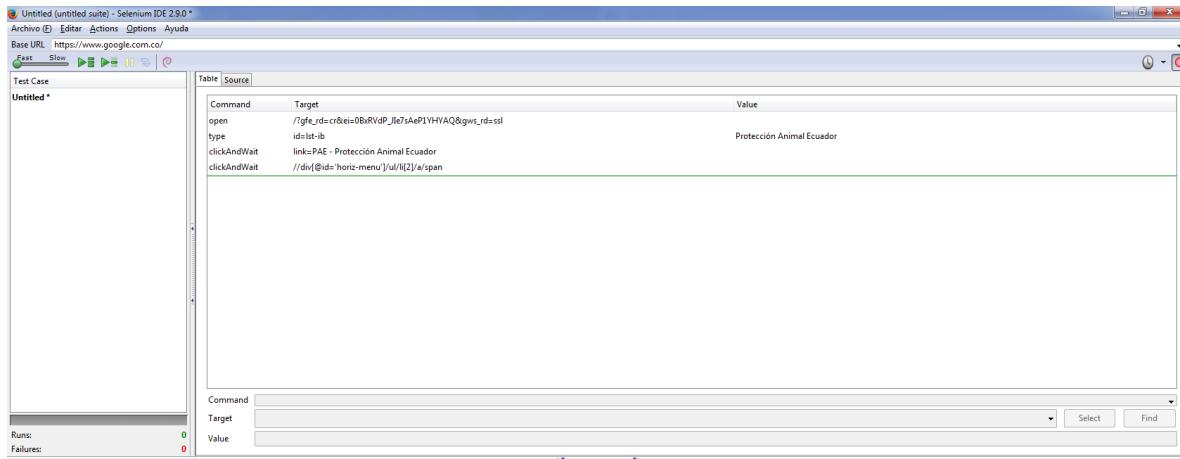


Figura 27

## Caso práctico

### Utilizando la Técnica de Partición de Equivalencia

Es una técnica de pruebas de software que divide los datos de entrada de una unidad de software en particiones de datos equivalentes desde el que se pueden derivar los casos de prueba.

Los casos de prueba están diseñados para cubrir cada partición al menos una vez. Esta técnica trata de definir los casos de prueba que descubren clases de errores, reduciendo así el número total de casos de prueba que debe ser desarrollada. Una ventaja de este enfoque es la reducción en el tiempo requerido para las pruebas de software debido a un menor número de casos de prueba.

**Sitio aplicativo web:** <http://pruebaideselenium.esy.es/Ventas.php>

**Tabla de verificación de registros:** <http://pruebaideselenium.esy.es/tabla-ventas.php>

Se tiene una aplicación web diseñada para la venta de productos en una distribuidora de alimentos.

Campos Objetivos de prueba.

- **Nombre de Cliente:** Que es una lista desplegable con los nombres de los clientes registradores previamente.
- **Descripción:** Almacena un texto no mayor a 12 caracteres.
- **Valor:** Puede tomar valores número de 0 a 999999.

The screenshot shows a web application window titled "Pruebas Selenium IDE". At the top is a navigation bar with a house icon and the title. Below it is a form titled "Registro Ventas". The form contains three input fields: "Nombre del Cliente" with a dropdown menu showing "Juan", "Descripción" (empty), and "Valor" (empty). To the right of the form is a small image of a shopping bag containing books. At the bottom of the form is a "Realizar Operación" button.

Figura 28

### Clases de Equivalencia:

Condición de entrada	Clases Validas	Clases invalidas
Nombre de Cliente Pauta 3.1 (Lista con elementos tratados igual)	1. Pedro o Juan	2. No seleccionar nada 3. Otro nombre que no está en la lista desplegable
Descripción Pauta 4 (Condición)	4. Cualquier cadena de caracteres alfanuméricos que este comprendida entre 0 y 12 caracteres.	5. Cualquier cadena de caracteres alfanuméricos que este comprendida con una longitud mayor a 12 caracteres. 6. No Ingresar ninguna descripción
Valor Pauta 1 (Rango)	7. $0 \leq Valor \leq 999999$	8. $10. 0 > Valor$ 9. $Valor > 999999$ 10. No ingresar valor

### Resumen de caso de prueba: Datos de Entrada y Valor Esperado:

No.	Clases de Equivalencia	Descripción	Valor	Nombre del cliente	Resultado Esperado (Mensaje)
1	1, 5, 7	Muy interesante la ubicación y longitud del mensaje	5228	Pedro	Error al almacenar la descripción
2	1, 4, 7	Efectivo	2540	Juan	Éxito de la operación
3	1, 3, 5, 6, 7, 9	Crédito	999999999 999999999	Pedro	Error al insertar la Valor
4	1, 4, 8	Crédito	-8989	Pedro	Error al insertar la Valor
5	2, 5, 7	Efectivo	5228	Juan	Éxito de la operación
6	3, 5, 6, 7	Efectivo	5228	Otoniel	Error al insertar la Valor
7	1, 4 , 6, 8	Efectivo	5228	Juan	Error al insertar la Valor
9					
10					

### Demostración de scripts de casos de prueba

#### Ejercicio no guiado

##### Ejercicio para Entregar.

Construir un scripts para realizar las pruebas de los requerimientos “adición de un nuevo usuario” y Eliminación de usuario.

Además debe incluir evidencia, caso de prueba y el resumen de caso de prueba.

Sitio aplicativo web: <http://pruebaideselenium.esy.es/Agregar-cliente.php>

The screenshot shows a web page titled "Pruebas Selenium IDE". A modal dialog box is open, titled "Agregar un nuevo cliente". It contains fields for "Nombre", "Telefono", "Celular", "Direccion", "Cedula", and "Comentario", each with a corresponding input field. Below these fields is a button labeled "Agregar Cliente".

Figura 29

Tabla de verificación de registros: <http://pruebaideselenium.esy.es/tabla-cliente.php>

The screenshot shows a web page titled "Pruebas Selenium IDE". A modal dialog box is open, titled "Tabla clientes". It contains a table with two rows of client data. At the top of the dialog, there is a message about MySQLi being deprecated. Below the table is a button labeled "Eliminar".

ID	Nombre	Teléfono	Celular	Dirección	Cédula	Comentario
2	Pedro	3654658	314 658 73 21	cuatro esquinas	1.856.452.515	Comprador regular
3	Juan	369 8524	321 258 96 32	Casona	1.148.442.358	Comprador Frecuente

Figura 30

### Requerimiento para la adición de un nuevo usuario

El sistema de permitir la creación de un cliente por medio de un formulario, el cual debe realizar el almacenamiento de los campos de:

- **Nombre:** con una longitud de carácter alfanuméricos no mayor a 20.
- **Teléfono:** con una longitud de carácter numérico no mayor a 99999999 dígitos.
- **Dirección:** con una longitud de carácter alfanuméricos no mayor a 2.
- **Celular:** con una longitud de carácter numérico no mayor a 9999999999 dígitos.
- **Cédula:** con una longitud de carácter numéricos no mayor a 20 dígitos.
- **Comentario:** con una longitud de carácter alfanuméricos no mayor a 128 caracteres.

## Referencias

**Notas de utilización de Selenio IDE**

<http://www.seleniumhq.org/>

**Selenium IDE - Release Notes**

<https://code.google.com/p/selenium/wiki/SelIDEReleaseNotes>